APPLICATION

FOR

UNITED STATES LETTERS PATENT

TITLE: PACKET PROCESSING

APPLICANT: ALLEN P. CHEN, JAYAKUMAR NATARAIAN, AND SIDDHARTH C. SHETH

CERTIFICATE OF MAILING BY EXPRESS MAIL

Express Mail Label No. EL688267869US

I hereby certify that this correspondence is being deposited with the United States Postal Service as Express Mail Post Office to Addressee with sufficient postage on the date indicated below and is addressed to the Commissioner for Patents, Washington, D.C. 20231.

November 29, 2000
Date of Deposit

Signature

Mike Augustine
Typed or Printed Name of Person Signing Certificate

PACKET PROCESSING

TECHNICAL FIELD

This invention relates to packet processing.

BACKGROUND

Networks (e.g., local area networks, wide area networks, intranets, extranets, or the Internet) typically include a backbone that transmits a communication signal (e.g., optical, electrical, or wireless) from a source to signal converters positioned at various points along this network backbone, which convert the data signal to a form usable by electrical data signal processing circuitry.

On packet switched networks, data is transported across these backbones in data chunks known as packets. Each packet has a destination address included in the packet's header. When these packets arrive at their destination, they are reassembled and provided to the destination device.

The signal processing circuitry that receives these data packets often has multiple data ports. Each port holds a data packet in queue for processing. During processing, each data port is polled to determine if it has a data packet available for processing. If such a packet is available, the packet is processed into a form usable by the device receiving the packet. Once this packet is fully processed, the polling process is once again initiated to determine if any other port has a packet available for processing.

Packets vary in length from small packets (approximately 64 bytes) to large packets (approximately 64 kilobytes). Whenever a large packet is received by a port and subsequently processed, the polling procedure is stalled until that large packet is fully processed. All other packets waiting to be processed on any other ports will be delayed until after the processing of the large packet is completed.

DESCRIPTION OF DRAWINGS

FIG. 1 is a diagrammatic view of a programmable intra-packet switching process;

FIG. 2 is a flow chart of a programmable intra-packet switching method;

FIG. 3 is a diagrammatic view of another programmable intra-packet switching process; and

FIG. 4. is a diagrammatic view of another programmable intra-packet switching process.

DETAILED DESCRIPTION

Referring to Fig. 1, a network 10 is used to transfer a data signal 12 from a source 14 to a network signal processing system 16. As stated above, network 10 can use any medium (e.g. electrical, optical, wireless, and so forth) and be of any type (e.g., a local area network, a wide area network, an intranet, an extranet, the Internet, Ethernet, Arcnet, Token Ring, packet-switched, or circuit-switched). Source 14 and network signal

processing system 16 can be incorporated into any device on a network, such as a printer, a personal computer, a router, a network server, a network interface card, or a cable modem.

Network signal processing system 16 includes a framer chipset 18, having one or more data ports 20_{1-N}. A typical example of framer chipset 18 is a PMC Sierra PMC5351 chip, available from Wyle Electronics of Irvine, California. While framer 18 is shown to have only four discrete ports (namely "A", "B", "C", and "D"), there could be more or less than four as needed. A signal converter may be utilized to convert data signal 12 received from network 10 into a signal usable by network signal processing system 16. For example, if network 10 is an optical network, an optical to electrical signal converter 19 may be incorporated between network 10 and framer chip 18 to convert the optical light-based signals into electrical-based signals.

Network signal processor circuit 22 interfaces with framer chipset 18, central processing unit (CPU) 24, and random access memory 26 using a high speed bus system, such as a local memory bus, a PCI bus, or a Utopia bus. A typical example of network signal processor circuit 22 is an Intel IXF6402 Asynchronous Transfer Mode/Packet Over Sonet (ATM/POS) processor.

In packet-switching networks (as opposed to circuit-switching networks), data signal 12 is transported in the form of packets 28_{1-N} of varyious sizes. As stated above, the size of packets 28_{1-N} can vary in size by orders of magnitudes (e.g., 64 bytes vs. 64 kilobytes).

As packets 28_{1-N} are received by framer chip 18, the individual data packets are stored in ports 20_{1-N} until these packets can be processed by network signal processor circuit 22. In order to balance the load between ports 20_{1-N} when a large packet is received for processing, a programmable intra-packet switching process 30 is executed on network signal processor circuit 22. Typically, intra-packet switching process 30 is stored on some form of non-volatile memory 32 (e.g., ROM, PROM, EEPROM).

Intra-packet switching process 30 includes a port polling process 34 for polling, in a systematic fashion, the individual data ports 20_{1-N} connected to network 10 to determine if any of these ports 20_{1-N} contains a data packet for processing. For example, if data packets 28_{1-N} include two packets (namely packets XX and XY), where XX is a relatively large packet (1500 bytes) and XY is a relatively small packet (approximately 100 bytes), these two packets (XX and XY) would be stored in any two data ports (e.g., ports "A" and "B" respectively of data ports 20_{1-N}). As stated above, port polling process 34 systematically polls ports 20_{1-N} to determine if any port has a data packet available for processing. In this example, polling process 10 would poll port "A" of ports 20_{1-N} and determine that port "A" indeed has a data packet (XX) available for processing.

Intra-packet switching process 30 includes a packet fragmentation process 36 which is responsive to port polling process 34 determining that port "A" of ports 20_{1-N} has a data packet (XX) available for processing. Packet fragmentation

process 36 fragments packet (XX) stored on port "A" of ports 20_{1-N} into one or more data cells 38_{1-N}, where each cell has a predefined size 31. This predefined size 31 is programmable and stored on non-volatile memory 32, where the size of the cells are chosen to maximize the processing efficiency of the system. While the size of these cells is user definable, a typical size is 53 bytes, which adheres to the Asynchronous Transfer Mode (ATM) guidelines. Data cells 38_{1-N} are stored on memory 26 for later processing (e.g. transmission, storage, and so forth).

Packet fragmentation process 36 continues fragmenting data cell (XX) until a user-defined number of cells 33 are generated or, alternatively, the data packet (XX) is fully fragmented. Again, this user-defined number of cells 33 is programmable and stored on non-volatile memory 32. The number can be chosen to maximize the processing efficiency of the system. A typical value for this cell number limit is three.

A user interface 40 allows a user 42 to program, among other things, the maximum number of cells 33 and the defined size 31 of the cells into which data packets 28_{1-N} are fragmented. User interface 40 can be any form of interface which allows the user to program these values, such as: DIP switches; LCD/LED displays; circuit board jumpers; or an HTTP-based web server in combination with an HTTP-based web browser (running on a remote computer 44).

A cell limit monitoring process 46 monitors the number of cells 38_{1-N} produced by packet fragmentation process 36 to determine if the user-defined number of cells 33 has been produced. As stated above, packet fragmentation process 36

continues fragmenting data packet (XX) until a user-defined number of cells 33 is produced, or until the data packet (XX) is fully fragmented. By setting a limit on the number of cells that can be produced before the port being serviced is switched, port loading can be controlled and reduced. Specifically, since the packet (XX) in port "A" is substantially larger than the packet (XY) in port "B", if fragmentation process 36 was allowed to continue to fragment data packet (XX) until the data packet was completely fragmented, a bottleneck would be created where the processing of all other ports would be stalled until the large packet (XX) in port "A" was completely fragmented. However, by limiting the number of cells that can be produced before another port is polled and another packet, if available, is processed, bottlenecks can be reduced and throughput increased.

Cell limit monitoring process 46 includes a cell limit port switching process 48 which is responsive to cell limit monitoring process 46 determining that the user defined number of cells 33 have been generated. If this occurs, cell limit port switching process 48 initiates port polling process 34 to determine if any other port contains a data packet which is available for processing. If cell limit port switching process 48, via port polling process 34, determines that another port has a data packet available for processing, cell limit fragmentation switching process 50 initiates packet fragmentation process 36 to fragment the data packet on the other port into cells having a user defined size 31. This fragmentation process continues until

the user defined number of cells 33 is generated, or the packet is fully fragmented.

Expanding on the example stated above, let's assume that user 42 programmed the user defined number of cells 33 to be twenty-five and programmed the cell size 31 to be fifty-three bytes. As stated above, two packets (namely packets XX and XY) are stored on ports "A" and "B" respectively, where XX is a relatively large packet (1500 bytes) and XY is a relatively small packet (approximately 100 bytes). Packet fragmentation process 36 would repeatedly fragment data packet XX into fifty-three byte cells. Cell limit monitoring process 46 would monitor the number of fifty-three byte cells generated by packet fragmentation process 36 until twenty-five cells have been produced. At this point, 1,325 bytes of the 1,500 byte packet (XX) have been fragmented into fifty-three byte cells. Cell limit port switching process 48 would initiate port polling process 34 to poll the other ports ("B", "C" and "D") to determine if any of these ports have a data packet available for processing. In this example, port polling process 34, upon polling port "B", would determine that a data packet (XY) is available for processing on port "B" and report this to cell limit port switching process 48. Cell limit fragmentation switching process 50, upon being informed that data packet (XY) is available on port "B", will initiate packet fragmentation process 36 and packet (XY) will be fragmented into fifty-three byte cells.

As the packet (XY) on port "B" is only one-hundred bytes in length, packet (XY) would be completely fragmented into only two

fifty-three byte cells. As will be explained below in greater detail, port polling process 34 would be once again initiated to determine if any other port (namely ports "C", "D" and "A") has a packet available for processing. Port polling process 34 would poll port "C" and find that no data packet was available. Port polling process 34 would then poll port "D" and, again, find that no data packet was available for processing. When port polling process 34 polls port "A", it would find the remainder of packet (XX), namely the 175 byte "chunk" not initially fragmented. Packet fragmentation process 36 would then fragment this packet "chunk" into four fifty-three byte cells.

Whenever cell limit monitoring process 46 determines that the user defined number of cells have been generated and cell limit port switching process 50 determines that another data packet is available for processing, the processing (i.e. fragmentation) of the first data packet is paused in lieu of processing (i.e. fragmenting) the second data packet. Accordingly, in order to facilitate subsequent processing of the first packet's "chunk" reminder, information concerning the first data packet must be stored for later retrieval.

Packet information storage process 52 is responsive to cell limit port switching process 48 determining that another port contains a data packet that is available for processing. Once it is determined that another port has a data packet available for processing, packet information storage process 52 stores various data elements 54 concerning the data packet currently being processed. These data elements 54 are stored on non-volatile

memory 32. Data elements 54 can include information concerning: the overall length of the data packet currently being processed; the length of the packet remainder to be processed; the length of the portion of the packet that has already been processed; a packet truncation indicator showing that the packet has not been completely processed; or a Packet-Over-Sonet (POS) header for use in optical networks. Accordingly, by storing data elements 54 on non-volatile memory 32, subsequent fragmentation of the remainder of a data packet (not completely fragmented due to cell number limit 33) can be easily achieved.

As stated above, once packet fragmentation process 36 starts to fragment a data packet, this fragmentation process 36 will continue until one of two things occur: (a) a user defined number of cells 33 is generated; or (b) the data packet being fragmented is completely fragmented. Intra-packet switching process 30 includes a packet completion monitoring process 56 which monitors the status of packet fragmentation process 36 to determine if the packet being processed (i.e. fragmented) has been completely fragmented into cells 38_{1-N}. Packet completion monitoring process 56 includes a packet completion port switching process 58 which is responsive to packet completion monitoring process 56 determining that the data packet currently being processed (i.e. fragmented) has been completely fragmented into cells 38_{1-N}. If this occurs, packet completion port switching process 58 initiates port polling process 34 to determine if any other port has a data packet available for processing. If packet completion port switching process 58, via port polling process 34,

determines that another port has a data packet available for processing, packet completion fragmentation switching process 60 initiates packet fragmentation process 36 to fragment the data packet on the other port into cells 38_{1-N}. This fragmentation process would continue until the user defined number of cells 33 are generated, or the packet is fully fragmented.

Now referring to Fig. 2, there is shown a programmable intra-packet switching method 100 in which a port polling process polls 102, in a systematic fashion, data ports connected to a network to determine which port, if any, contains a data packet available for processing. A packet fragmentation process fragments 104 the available data packet into at least one data cell having a defined size, where this fragmentation continues until a user-defined number of cells are generated.

A cell monitoring process monitors 106 the number of data cells produced to determine if the user defined number of cells have been generated. If so, a cell limit port switching process initiates 108 the polling process to determine if any other port contains a data packet available for processing. If such a data packet is available for processing, a packet information storage process stores 110 at least one data element concerning the data packet currently being processed, where the data element allows for subsequent processing of the remainder of the data packet currently being processed. Further, if such a data packet is available for processing, a cell limit fragmentation switching process initiates 112 the packet fragmentation process to fragment the data packet on the other port into at least one data

cell having a defined size, where the packet fragmentation process continues fragmenting the data packet on the other port into data cells until the user-defined number of cells are generated.

A packet completion monitoring process determines 114 if the data packet being processed has been fully fragmented into at least one data cell. If it is determined that the data packet has been fully fragmented, a packet completion port switching process initiates 118 the polling process to determine if any other port contains a data packet available for processing. If it is determined that another port contains a data packet for processing, a packet completion fragmentation switching process initiates the fragmentation process to fragment the data packet on the other port into at least one data cell having a defined size, where the packet fragmentation process continues fragmenting the data packet on the other port into data cells until the user-defined number of cells are generated.

Once these cells are generated, processing can continue via processing circuitry (not shown) which allows for the subsequent manipulation or transmission of these cells on a cell-by-cell basis, as opposed to a packet-by-packet basis.

Now referring to Fig. 3, there is shown a computer program product 200 which functions within a network signal processor circuit. Computer program product 200 resides on a computer readable medium 202 which has a plurality of instructions 203 stored thereon. When executed by processor 204, instructions 203 cause processor 204 to poll, in a systematic fashion, a plurality

of data ports connected to a network to determine which port, if any, contains a data packet available for processing. Computer program product 200 fragments the available data packet into at least one data cell having a defined size, where this fragmentation continues until a user-defined number of cells are generated.

Typical embodiments of computer readable medium 202 are: hard drive 210; optical drive 212; random access memory 214; tape drive 216; RAID array 218; and read only memory 220.

Now referring to Fig. 4, there is shown a processor 300 and memory 302 configured to poll 304, in a systematic fashion, a plurality of data ports connected to a network to determine which port, if any, contains a data packet available for processing. Processor 300 and memory 302 fragment 306 the available data packet into at least one data cell having a defined size, where this fragmentation continues until a user-defined number of cells are generated.

Processor 300 and memory 302 may be incorporated into a personal computer 308, a programmable logic controller 310, a single board computer 312, or an array of network servers 314.

Other implementations are within the scope of the following claims.